International Journal For Technological Research In Engineering
Volume 4, Issue 4, December-2016 *ISSN (Online): 2347 - 4718*

MAPREDUCE FOR BIG DATA APPLICATIONS TO ENHANCE THE NETWORK TRAFFIC PERFORMANCE

Kagithapu Navyajyothi[1], Dr. Kasukurthi Venkata Rao[2]
[1]M. Tech Scholar, [2]Professor
Department Of Computer Science and System Engineering,College of Engineering, Andhra University Visakhapatnam, Andhra Pradesh, India.

Abstract: MapReduce is a scheme for processing and managing large scale data sets during a distributed cluster, which has been used for applications such as document clustering, generating search indexes, access log analysis, and numerous other forms of data analytic. In existing system, a hash function is used to partition intermediate data among reduce tasks. During this project the system proposed a decomposition-based distributed algorithm to deal with the large-scale optimization problem for large data application and an online algorithm is additionally designed to adjust data partition and aggregation in a dynamic manner. Network traffic price under both offline and on-line cases is significantly reduced as demonstrated by the extensive stimulation results by the various proposals considered and used.

Index Terms: Map phase, Reduce phase, shuffle phase, Aggregator; distributed system;

I. INTRODUCTION

Map-Reduce have appeared as the terribly popular calculative framework for large data processing appropriate to its simple programming model and automatic parallel execution. Map-Reduce and its open source fulfillment Hadoop are utilized by many massive companies, like Yahoo!, Google and Facebook, for different massive data applications, like machine learning, bioinformatics, and cyber security Map-Reduce separate a task into two main phases, namely map and reduce which in bend are transported out by several reduce tasks and map tasks, respectively. Within the map phase, map computations are started in parallel to change the master input divides into intermediate data during a form of key-value pairs. These key-value pairs are saved on native machine and formed into several data partitions and perform reduce task. Within the reduce part, each reduce computation takes its own share of data partitions from all map work to develop the final result. During a shuffle step compute between map and reduce part. In this step, the data given by the map phase are ordered, split and transmitted to the exact machines performance the reduce part. The final resulting network traffic from all map computation to any or all reduce computation can reason volume of network traffic, impressive a force on the ability of data analytical applications. In , intermediate data are shuffled agreed to a hash function in Hadoop, which would occurred to giant network traffic reason it neglected configuration and data size related to every key. However it this hash function very slow process the major advantage of Map-Reduce is that it's easy to scale processing over multiple Calculating nodes

The computing takes place on nodes with data on local disks that reduces the network traffic .Map-Reduce program executes in three phases, namely map phase, shuffle phase, and reduce phase. In Existing system, a decomposition-based distributed algorithm is projected to deal with the large-scale optimization drawback for large data application and an online algorithm is additionally designed to regulate data partition and aggregation during a dynamic manner. In this work we collectively consider data partition and aggregation for a Map-Reduce job with associate degree objective that's to minimize the whole network traffic. In our projected system, we introduced new Meta Heuristic approach to perform the load balancing. The optimization algorithm is a probabilistic technique for resolution computational problems which might be reduced to finding good paths through graphs. The optimization is random optimization search technique which will be used for allocating the incoming jobs to the virtual machines.

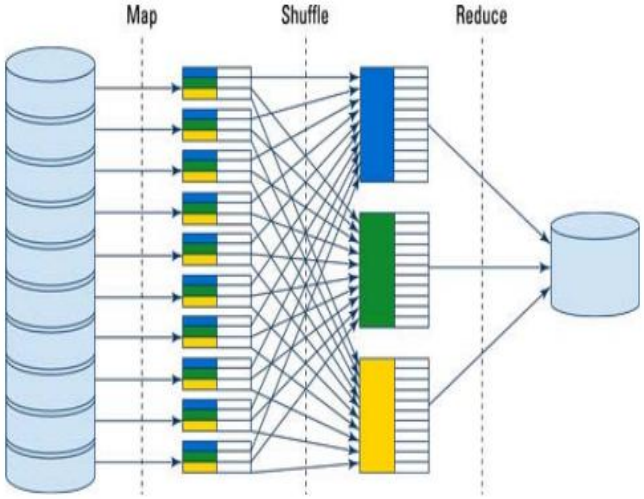


Figure 1: Mapper & Reducer task using Aggregation

If we mixture the data of a similar keys before sending them over the top switch, as shown in Figure 1., the network traffic will be reduced. During this paper, we collectively consider data partition and aggregation for a Map-Reduce job with an objective that's to reduce the total network traffic. Above all, we propose a distributed algorithm for large data applications by decomposing the initial large-scale drawback into several sub issues which will be solved in parallel. An online algorithmic rule is developed to perform with the data partition and aggregation during a dynamic manner. Finally, simulation results describe that our offer will reduce network traffic value in each offline and on-line cases.

II. RELATED WORK

Much present work depends on Map-Reduce task to get better by optimizing its data transforming. Blancaeal. have asked the question of optimizing network usage will improve to better system task and got that high network access and low network congestion should be performed at the same time for a job with good method. Cloud task programming supported antcolony optimization Medhat al. described this paper, a cloud computation programming policy depends on ant Colony optimization algorithm compared with completely different programming algorithms like First come first Served (FCFS) and Round-Robin(RR), has been given. The most goals of those algorithms are minimizing set of computation. ACO israndom optimization search designed which will be utilized for assigning the incoming jobs to the virtual machines. Algorithms are calculated using cloud-sim toolkit package. Examining results exhibited that cloud compute scheduling based on ACO calculated FCFS and RR algorithms. A Task programming Algorithm supported Genetic algorithm and optimization in Cloud Computing shekharal., determined the paper we offer detail plan about Genetic algorithm and its many various proposed for task programming in cloud environment and idea of GA based scheduler is described within which population is given by max and min by that develop span can be reduced and cargo of resources are often balanced. Job scheduling based on ant colony optimization in cloud environment al., represented calculated victimization the imitated process times for a cloud environment. Cloud environment is performed dynamically so, prophosy primarily based analysis isn't possible and execution controlling of any application in cloud computing is very much necessary. This paper proposes a programming algorithm for correct use of the resources and to reduce collision in cloud environment. Ant colony optimization for resource-strained project scheduling Daniel al., determined that paper optimization approach for the resource on strained project scheduling problem is presented. Combinations of two secretion analysis methods are utilized by the ants to find new solutions. Wetested our optimization algorithm on a collection of huge benchmark problems from the PSPLIB. Compared too many alternative heuristics for the RCPSP including genetic algorithms, simulated annealing, search, and different sampling ways our algorithm performed best on the average. For a few take a look at instances the algorithm was ready to find new best solutions. An ant colony improvement approach for the resource-strained project scheduling problem is showed. Joined of two pheromone execution ways are used by the ants to get new solutions. Calculated annealing, search, and different sampling ways our algorithm examined best on the average. For some access instances the algorithm was able to provide new best solutions.

III. FRAME WORK

Map Phase and Reduce phase like models are widely used to process “Big Data”. Applications based on suchmodels place heavily data-dependency or communication on Virtual Machines; therefore network traffic becomes the bottleneck of jobs. The following three phase sof data exchange within the execution method of an application based on Map Reduce model. This paper point to the provisioning a virtual cluster according to the position relationship between Virtual Machines therefore on decrease the network traffic and improve the performance of Map phase and Map Reduce phase like applications rather than modifying the job scheduling methods or Virtual Machine configurations. By optimizing the architecture of virtual clusters, cloud users will get a more efficient platform with an equivalent resource request and value, and cloud providers will get a better resource utilization ratio. The most contributions of this paper area unit summarized below during this technique to measure the affinity by process the distance of a virtual cluster the shorter the distance, the nearer the virtual cluster. The shortest distance drawback is presented to obtain the closest virtual cluster to resolve the shortest distance problem by formulating it into a number linear programming. A heuristic Virtual Machine placement rule is put forward to provision a virtual cluster. It is designed for MapReduce applications to improve the shuffle speed and stimulate the execution. It is also optimize the virtual cluster from the global, i.e., provisioning virtual clusters for a request queue rather than one request. The web heuristic Virtual Machine placement algorithm and the global optimization algorithm are compared by simulations. The former has lower time complexity whereas the latter arrival shorter average distance for multiple requests to analyze the performance of our approach through experiments. In the experiment, describe the support different virtual cluster architectures to check different MapReduce applications. Two metrics of application runtime and cluster compatibility show the efficiency of virtual cluster optimization. The following figure 2 shown the simple Map reduce task using key with aggregation.

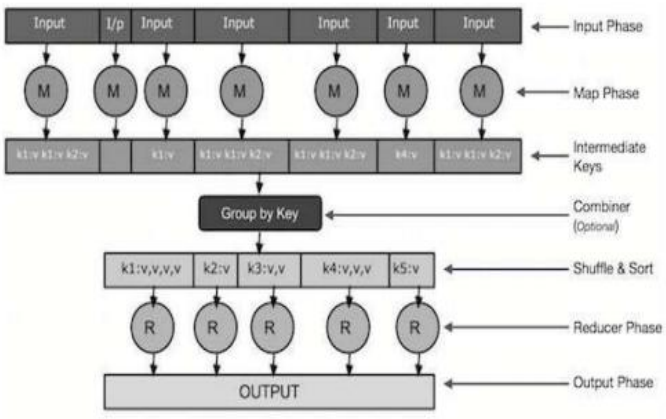


Figure 2: MapReduce Processing

The system proposed a distributed algorithm for bigdata applications by decomposing the original large-scale problem into several sub issues which will besolved in parallel The system investigate network traffic reduction within Map reduce jobs by jointly exploiting traffic-aware intermediate data partition and data aggregation among multiple map tasks. It offers computers as physical or more often as virtual machines. A cluster of virtual machines, virtual cluster, is often requested as a platform for users to

run parallel or distributed applications like Map reduce and dryad applications. So as to get high throughput, fast response, load balance, low cost, and low worth, many topics on virtual machines configuration, virtual machines placement, virtual machines consolidation, and virtual machines migration are explored. The network topology of a virtual cluster includes an important impact on the execution of applications running on it because the physical nodes where virtual machines are located will be linked in several ways. For example, some nodes are located in the same rack whereas others in several racks through a slow link. Map and reduce tasks may partially overlap under some cases however the execution is to increase system throughput, and it is difficult to estimate system parameter set a high accuracy for big data applications. An online algorithm to dynamically adjust data partition and aggregation during the execution of map and reduce tasks is so motivated. The basic plan of this algorithm is to postpone the migration operation till the cumulative traffic cost exceeds a threshold. Another is on-line algorithm which is additionally designed to deal with the info partition and aggregation in a very dynamic manner. The simulation results finally demonstrated recommend that our proposals will significantly reduce network traffic value in both offline and on-line cases.

IV. ALGORITHM

In case of "Big data" the major challenges with high efficient algorithms, e.g., branch and bound, fast off-the-shelf solvers leading to a large-scale optimization problems. Therefore we developed a distribution algorithm to solve the problem on multiple machines in a parallel manner. Our basic idea is to decompose the original large-scale problem into several distributively solvable sub problems that are coordinated by a high-level master problem. To achieve this objective, we first introduced an auxiliary variable z_j^p such that our problem can be equivalently formulated as

Table 1

Notations	Description
N	A set of physical machines
d_{xy}	Distance between two machines
M	A set of map tasks in map layer
R	A set of reduce tasks in reduce layer
A	A set of map tasks in map layer
P	A set of nodes in aggregation
A_i	A set of neighbors of mapper i∈ M
x_{ij}^p	Binary variable denoting whether mapper i∈ M sends data of key p∈P to j∈A
f_{ij}^p	Traffic for key p∈ P from mapper to i∈M to node j∈ A
M_j	A set of neighboring nodes of j∈ A
z_j	Binary variable indicating if an aggregator is placed on machine j∈N
y_k^p	Binary variable denoting whether data of key p∈ P is processed by reducer k∈R
g_{jk}^p	The network traffic regarding key p∈ P from node j∈A to reducer k∈R
v_{jk}^p	Lagrangian multiplier
z_j^p	An auxiliary variable
$C_M(t)$	Total migration cost at time slot t

$$\min \sum_{p\in P}\left(\sum_{i\in M}\sum_{j\in A_i} f_{ij}^p d_{ij} + \sum_{j\in A}\sum_{k\in R} g_{jk}^p d_{jk}\right) \quad (1)$$

Subjected to: $x_{ij}^p \le z_j^p$, j∈ N, i∈ M_j, p ∈ P, (2)
$z_j^p = z_j$, j∈ N, p ∈ P,
The corresponding Lagrangian is as follows

$$\begin{aligned} L(v) &= \sum_{p\in P} C^p + \sum_{j\in N}\sum_{p\in P} v_j^p (z_j - z_j^p) \\ &= \sum_{p\in P} C^p + \sum_{j\in N}\sum_{p\in P} v_j^p z_j - \sum_{j\in N}\sum_{p\in P} v_j^p z_j^p) \\ &= \sum_{p\in P}(C^p + \sum_{j\in N} v_j^p z_j^p) + \sum_{j\in N}\sum_{p\in P} v_j^p z_j^p) \end{aligned} \quad (3)$$

Given v_j^p, the dual decomposition results in two sets of the subproblems: intermediated data partition and aggregator placement. The subproblem of data partition for each key p∈ P is as follows:

SUB_DP:

$$\min \quad (C^p - \sum_{j\in N} v_j^p z_j^p)$$

These problems regarding different keys can be distributed solved on multiple machines in a parallel manner.
The subproblem of aggregator placement can be simply written as:
SUB_AP:

$$\min \quad \left(\sum_{j\in N}\sum_{p\in P} v_j^p z_j\right)$$

The values of v_j^p are updated in the following master problem:

$$\min\ L(V) = \sum_{p\in P} \hat{C}^p + \sum_{j\in N}\sum_{p\in P} v_j^p \hat{z}_j - \sum_{j\in N}\sum_{p\in P} v_j^p \hat{z}_j^p$$
Subject to: $v_j^p \ge 0, \forall j \in A, p \in p,$

Where$\hat{C}^p$, $\hat{z}_j^p$ and $\hat{z}_j$ are optimal solutions returned by subproblems.

Algorithm 1 Distributed Algorithm
1: set t =1, and $v_j^p (j \in A, p \in P)$ to arbitrary non negative values;
2: **for** t< T do
3: Distributively solve the subproblems SUB_DP and

SUB_AP on multiple machines in a parallel manner;
4: update the values of v_j^p with the gradient method, and send the results to all subproblems
5: set t=t+1;
6: **end** for
Since this objective function of the master problem is differentiable, it can be solved by the following gradient method.

$$v_j^p(t+1) = \left[v_j^p + \xi\left(\hat{z}_j\left(v_j^p t\right) - \hat{z}_j^p\left(v_j^p(t)\right)\right)\right]^+$$

Where t is the iteration index, ξ is a positive step size, and '+' denotes the projection onto the nonnegative orthants.

ONLINE ALGORITHM:
In practice map and reduce task may partially overlap in execution to increase system throughput, and it is difficult to estimate the system parameters at high accuracy for "Big data" applications. Therefore in this work online algorithm is proposed.
The execution of a MapReduce jon into several time slots with a length of several minutes or an hour. We let $m_j^p(t)$ and $\alpha_j(t)$ denote the parameters collected at time slot t with no assumption about their distributions. As the job is running, an existing partition and aggregation scheme may not be optimal anymore under current $m_j^p(t)$and$\alpha_j(t)$. To reduce the cost, we may need to migrate a aggregator from machine j to $j^{'}$ with a migration cost $\Psi_{jj'}$. Meanwhile, the key assignment among reducers is adjusted. When we let reducer $k^{'}$ process the data with key p instead of reducer k that is currently in charge of this key. We use function

$$\Phi_{kk'}\left(\sum_{r=1}^{t}\sum_{j\in A}\sum_{k\in R} g_{jk}^p(\tau)\right)$$

The total migration cost can be calculated as

$$C_M(t) = \sum_{k,K'\in R}\sum_{p\in P} y_k^p(t-1)y_{k'}^p(t)\Phi_{kk'}.$$

$$\left(\sum_{r=1}^{t}\sum_{j\in A}\sum_{k\in R} g_{jk}^p(\tau)\right)) + \sum_{j,j'\in N} z_j(t-1)z_{j'}(t)\Psi_{jj1}$$

Our objective is to minimize the overall cost of traffic

Algorithm 2 Online Algorithm
1: t=1 and $\hat{t} = 1$;
2: solve the OPT_ONE_SHOT problem for t=1;
3: **while** t ≤ T **do**
4: **if**$\sum_{\tau=\hat{t}}^{t}\sum_{p\epsilon P} C_t^p(\tau) > \gamma C_M(\hat{t})$then
5: solve the following optimization problem

$$min\sum_{p\in P} C^p(t)$$

6: **if** the solution indicates a migration event then
7: conduct migration according to the new solution
8: $\hat{t} = t$;
9: update $C_M(\hat{t})$
10: **end** if
11: **end** if
12: t = t+1;
13: **end** while

And migration over a time interval [1, T], i.e.,

$$\min\sum_{t=1}^{T}\left(C_M(t) + \sum_{p\in P} C^p(t)\right)$$

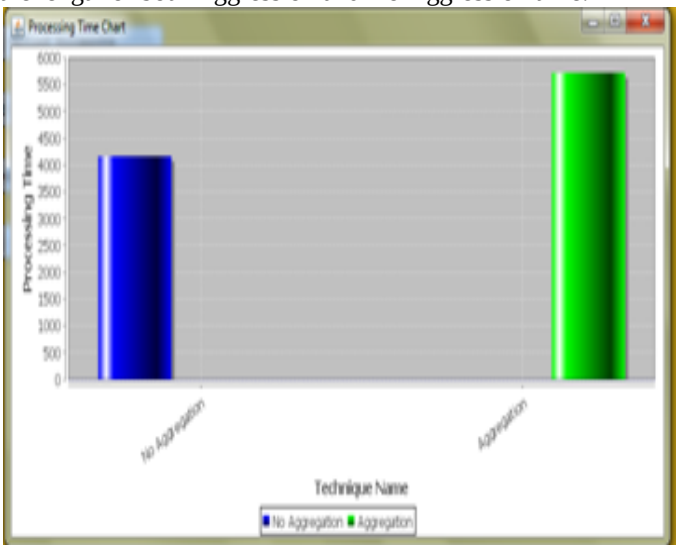
An intuitive method to solve the problem above is to divide it into T one-shot optimization problems:
OPT_ONE_SHOT:

$$\min C_M(t) + \sum_{p\in P} C^p(t)$$

Unfortunately, the algorithm of solving above one-shot optimization in each time slot based on the information collected in the previous time slot will be fat from optimal because it may lead to frequent migration events. Therefore for we design an online algorithm whose basic idea is to postpone the migration operation until the cumulative traffic cost exceeds a threshold.

V. EXPERIMENTAL RESULTS

In this paper we have described a map reduce frame work used for parallel processing large amount of data. Although there are many existing works which have been focused on the traffic reduction, they did not do well in the map reduce paradigm. Those works focused mainly on the map and reduce phases instead of concentrating on shuffle phase. They did not attempt to reduce the data traffic for the improvement of the network traffic and the data cost. Although the existing works focused on the traffic improvement they utilized the large number of keys in the system to make a process more complex. In this paper, we have implemented an efficient system for map reduces jobs by data partition and aggregation for the big data applications.
In the below chart we can observe that difference between the length of both Aggression and No Aggression time.



We can observe that Aggression length is higher than No Aggression length. The difference will be shown in the sense of Processing Time. Through our implementation we have implemented an efficient system for map reduces jobs by data partition and aggregation for the big data applications so we can consider that improving the network traffic performance and the data cost.

VI. CONCLUSION AND FUTURE WORK

We propose a three-layer model for this problem and formulate it as a mixed-integer nonlinear problem, which is then transferred into a linear form that may be solved by mathematical tools. To deal with the large-scale formulation due to massive data, we design a distributed algorithm to resolve the problem on multiple machines. The joint optimization of intermediate data partition and aggregation in Map Reduce to minimize network traffic cost for large data applications is studied. It is more; we plan to extend our rule to handle the Map Reduce job in an online manner when some system parameters are not given. Finally, we are going to conduct extensive simulations to evaluate our proposed rule under each offline cases and on-line cases. The simulation results demonstrate that our proposals will effectively reduce network traffic cost.

REFERENCES

[1]. J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," Communications ofthe ACM, vol. 51, no. 1,pp. 107–113, 2008.

[2]. W. Wang, K. Zhu, L. Ying, J. Tan, and L. Zhang, "Map task scheduling in mapreduce with data locality: Throughput and heavy-traffic optimality," in INFOCOM, 2013 Proceedings IEEE. IEEE, 2013, pp. 1609–1617.

[3]. F. Chen,M. Kodialam, and T. Lakshman, "Joint scheduling of processing and shuffle phases in mapreduce systems," in INFOCOM, 2012 ProceedingsIEEE. IEEE, 2012, pp. 1143–1151.

[4]. Y. Wang, W. Wang, C. Ma, and D. Meng, "Zput: A speedy data uploading approach for the hadoop distributed file system," in Cluster Computing (CLUSTER), 2013 IEEE International Conference on. IEEE, 2013, pp. 1–5.

[5]. T. White, Hadoop: the definitive guide: the definitive guide. " O'Reilly Media, Inc.", 2009.

[6]. S. Chen and S. W. Schlosser, "Map-reduce meets wider varieties of applications," Intel Research Pittsburgh, Tech. Rep. IRP-TR-08-05,2008.

[7]. S. Venkataraman, E. Bodzsar, I. Roy, A. AuYoung, and R. S.Schreiber, "Presto: distributed machine learning and graph processing with sparse matrices," in Proceedings of the 8th ACM European Conference onComputer Systems. ACM, 2013, pp. 197–210.

[8]. A. Matsunaga, M. Tsugawa, and J. Fortes, "Cloudblast: Combining mapreduce and virtualization on distributed resources for bioinformatics applications," in eScience, 2008. EScience'08. IEEEFourth International Conference on. IEEE, 2008, pp. 222–229.

[9]. J. Wang, D. Crawl, I. Altintas, K. Tzoumas, and V. Markl, "Comparison of distributed data-parallelization patterns for big data analysis: A bioinformatics case study," in Proceedings of the Fourth InternationalWorkshop on Data Intensive Computing in the Clouds(DataCloud), 2013.

[10]. R. Liao, Y. Zhang, J. Guan, and S. Zhou, "Cloudnmf: A mapreduce implementation of nonnegative matrix factorization for largescale biological datasets," Genomics, proteomics & bioinformatics, vol. 12, no. 1, pp. 48–51, 2014.

[11]. G. Mackey, S. Sehrish, J. Bent, J. Lopez, S. Habib, and J. Wang,"Introducing map-reduce to high end computing," in Petascale Data Storage Workshop, 2008. PDSW'08. 3rd. IEEE, 2008, pp. 1–6.

[12]. W. Yu, G. Xu, Z. Chen, and P. Moulema, "A cloud computing based architecture for cyber security situation awareness," in Communications and Network Security (CNS), 2013 IEEE Conference on. IEEE, 2013, pp. 488–492.

[13]. F. Ahmad, S. Lee,M. Thottethodi, and T. Vijaykumar, "Mapreduce with communication overlap,"pp. 608–620, 2013.

Kagithapu Navyajyothi currently pursuing her Post Graduation in Computer Science and Systems Engineering, College of Engineering, Andhra University, Visakhapatnam, AP, India. Her area of interest includes Big data Analytics.

K.Venkata Rao is currently working as a Professor, in Computer Science and Systems Engineering, College of Engineering, Andhra University, Visakhapatnam, AP,India. He has more than 15 years of experience in teaching field. His research field includes Image Processing and Bigdata Analytics.

